Revisiting Clustered Microarchitecture for Future Superscalar Cores: A Case for Wide Issue Clusters

P. Michaud, A. Mondelli, A. Seznec

2016 Conference
January 18-20, 2016, Prague, Czech Republic

Motivations

Multicore system

benefits applications with thread-level

Reality

*most applications have low thread-level parallelism

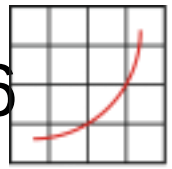
Table of contents

Quantity → Potential IPC gain ?

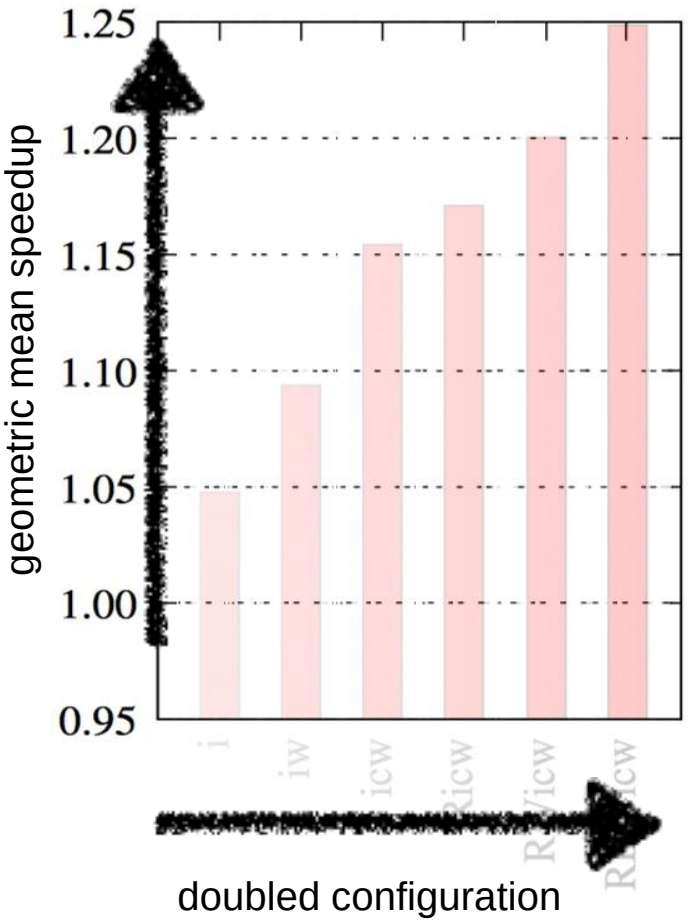
Quality → In which way we should proceed ?

Solutions → Future of superscalar architecture

Simulation Environment

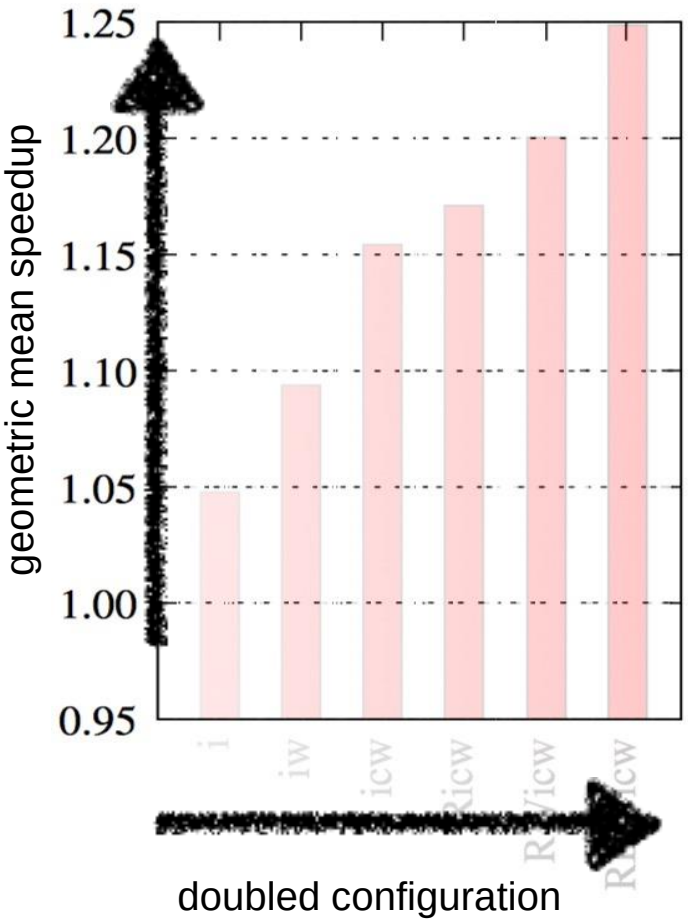
- *in-house simulator based on* Pin
- *benchmarks used* SPEC CPU 2006 
- *1 billion insts simulated (20 samples x 50 millions)*
- *aggressive Haswell-like baseline* intel Haswell

Future Potential IPC Gain



- *7 different parameters*
- *128 configurations*
- *highest speedup per configuration*

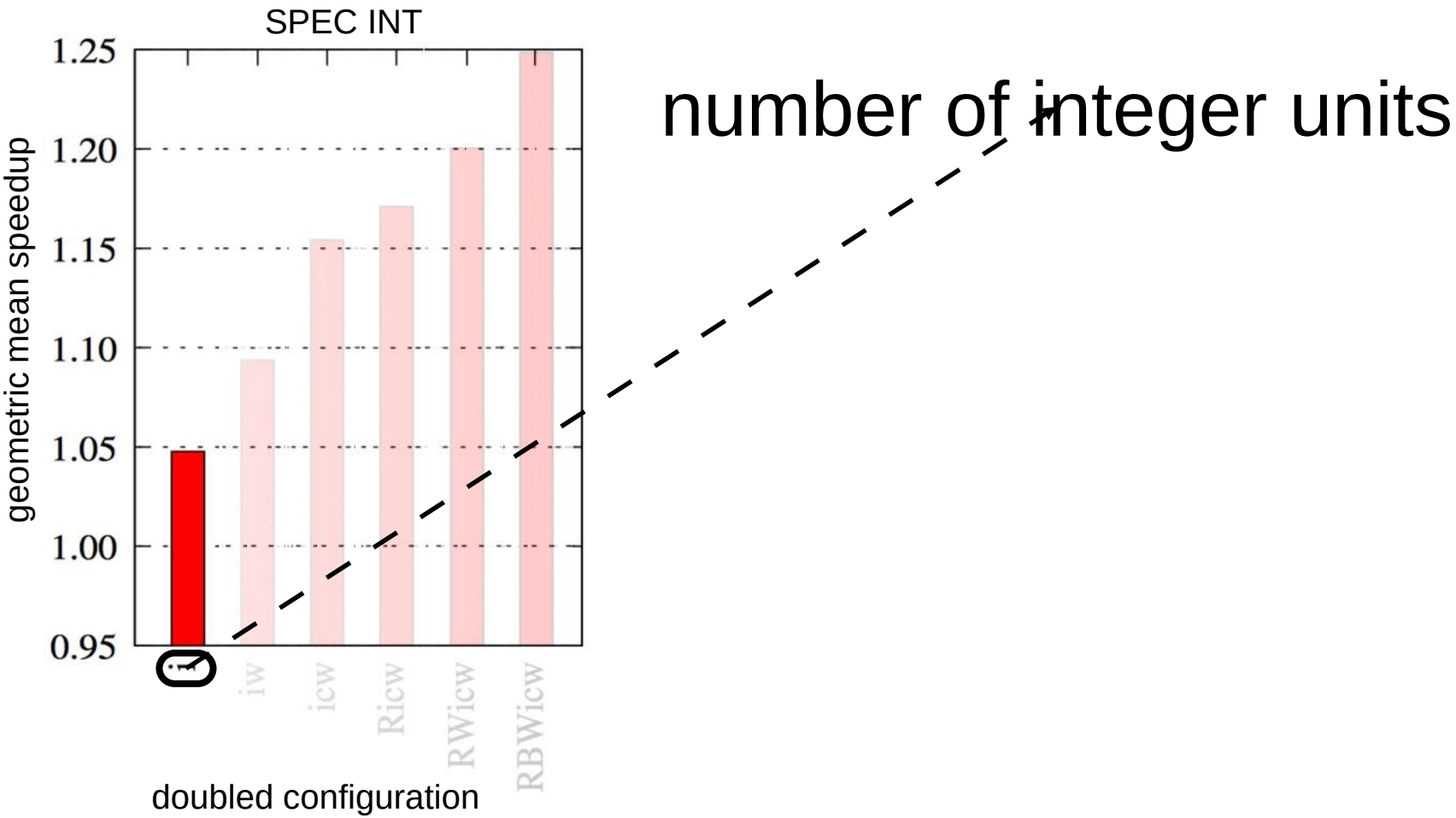
Future Potential IPC Gain



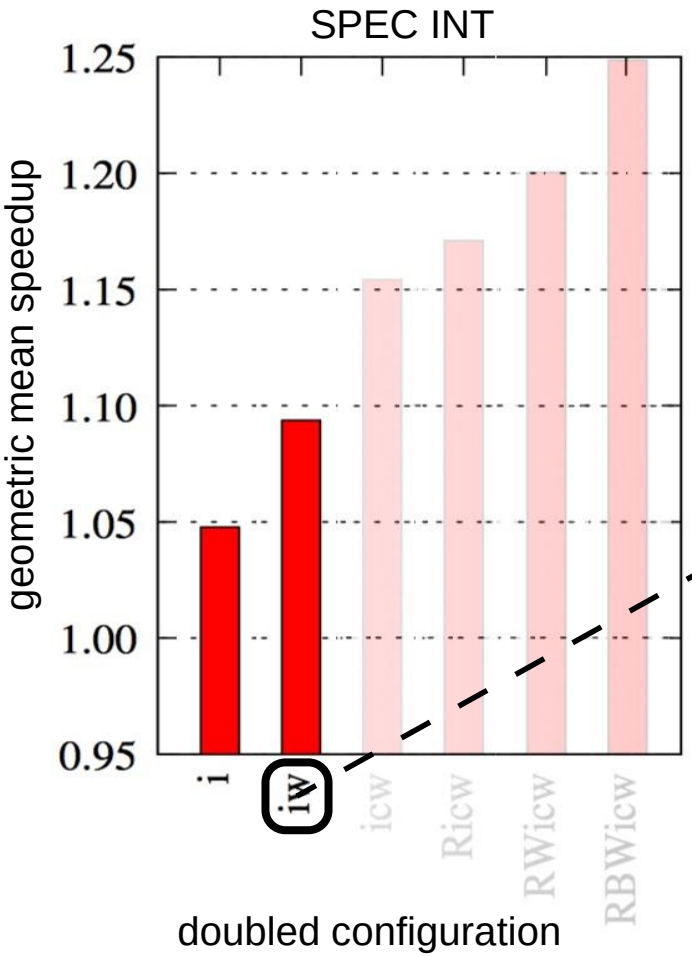
- *7 different parameters*
- *128 configurations*
- *highest speedup per configuration*

The extra complexity can be introduced incrementally

Future Potential IPC Gain

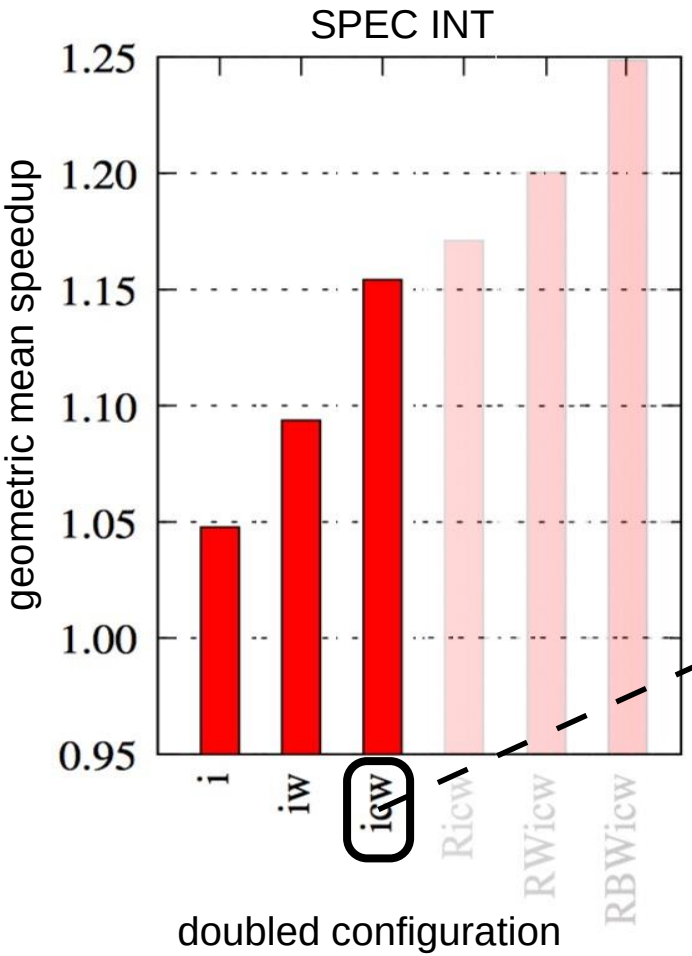


Future Potential IPC Gain



number of integer units
+
front-end width

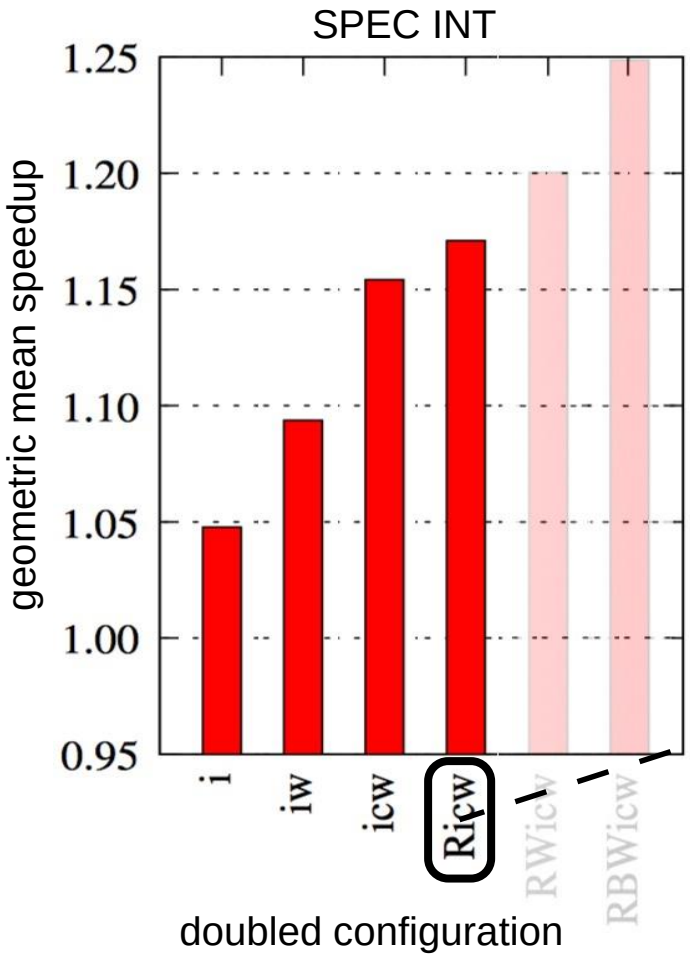
Future Potential IPC Gain



number of integer units
+
front-end width
+
memory parameters*

*__memory__ = load issue + DL1 write ports

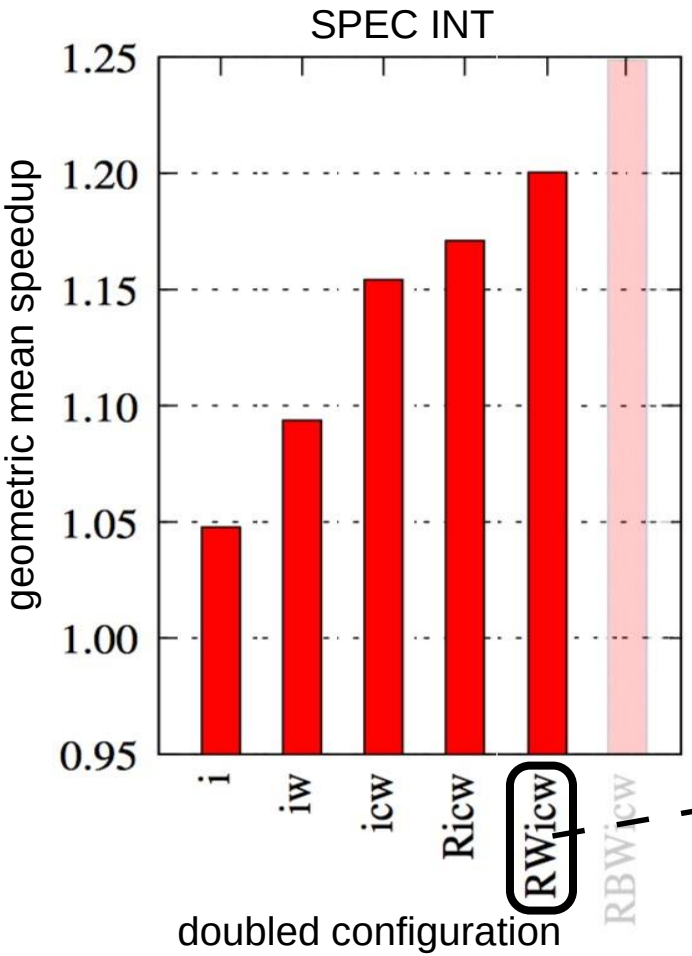
Future Potential IPC Gain



number of integer units
+
front-end width
+
memory parameters*
+
physical register file

*__memory__ = load issue + DL1 write ports

Future Potential IPC Gain

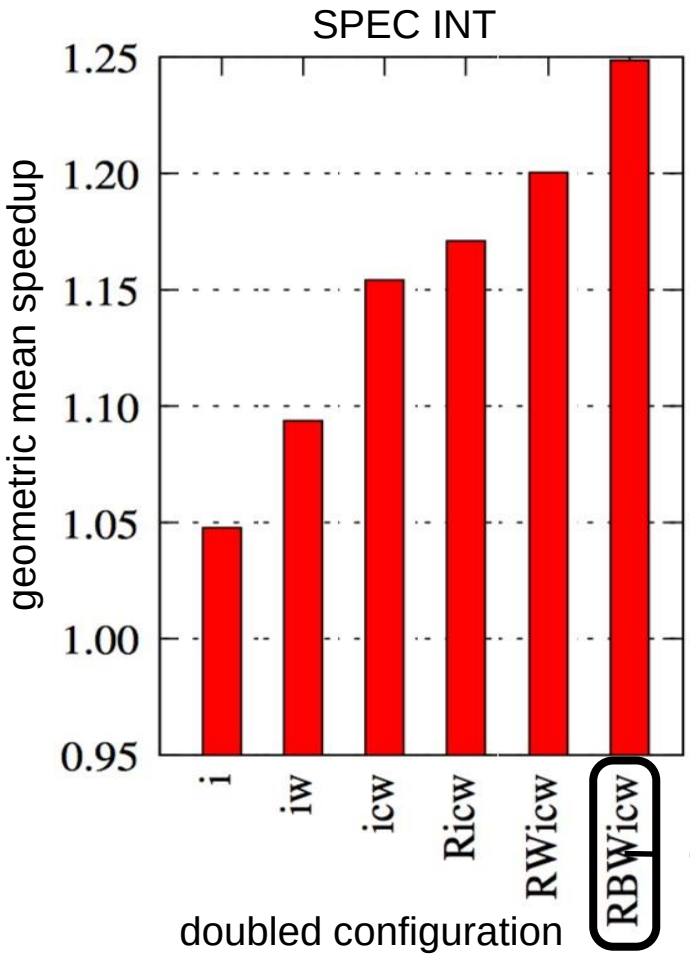


number of integer units
+
front-end width
+
memory parameters*
+
physical register file
+
instruction window*

*__memory__ = load issue + DL1 write ports

*__instruction window__ = ROB + L/S queue + LFST + MSHR

Future Potential IPC Gain



number of integer units
+
front-end width
+
memory parameters*
+
physical register file
+
instruction window*
+
issue buffer

*__memory__ = load issue + DL1 write ports

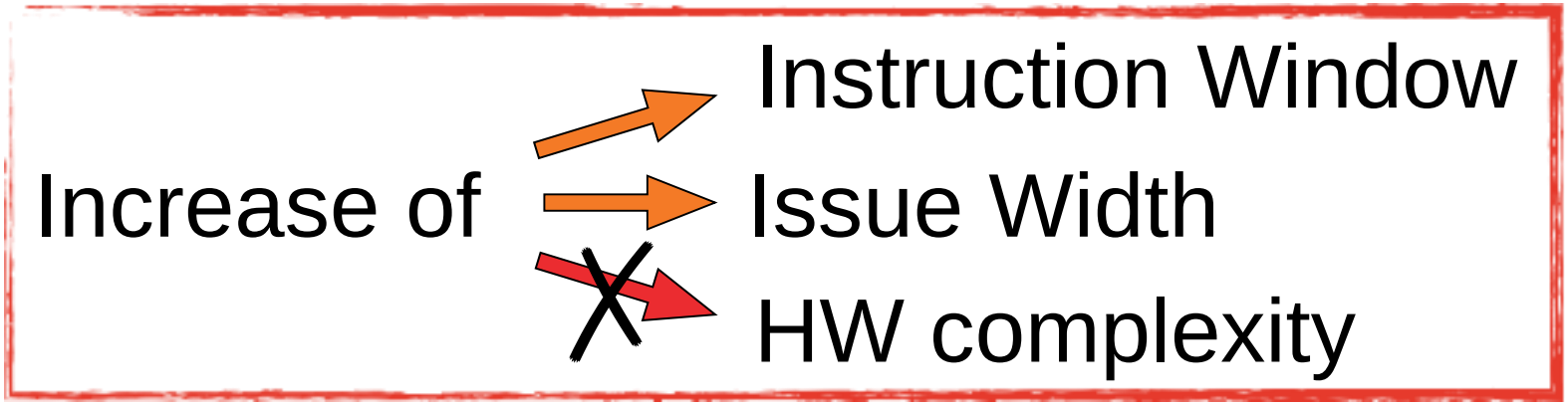
*__instruction window__ = ROB + L/S queue + LFST + MSHR

Future of Superscalar

Increase of

- Instruction Window
- Issue Width
- ~~HW complexity~~

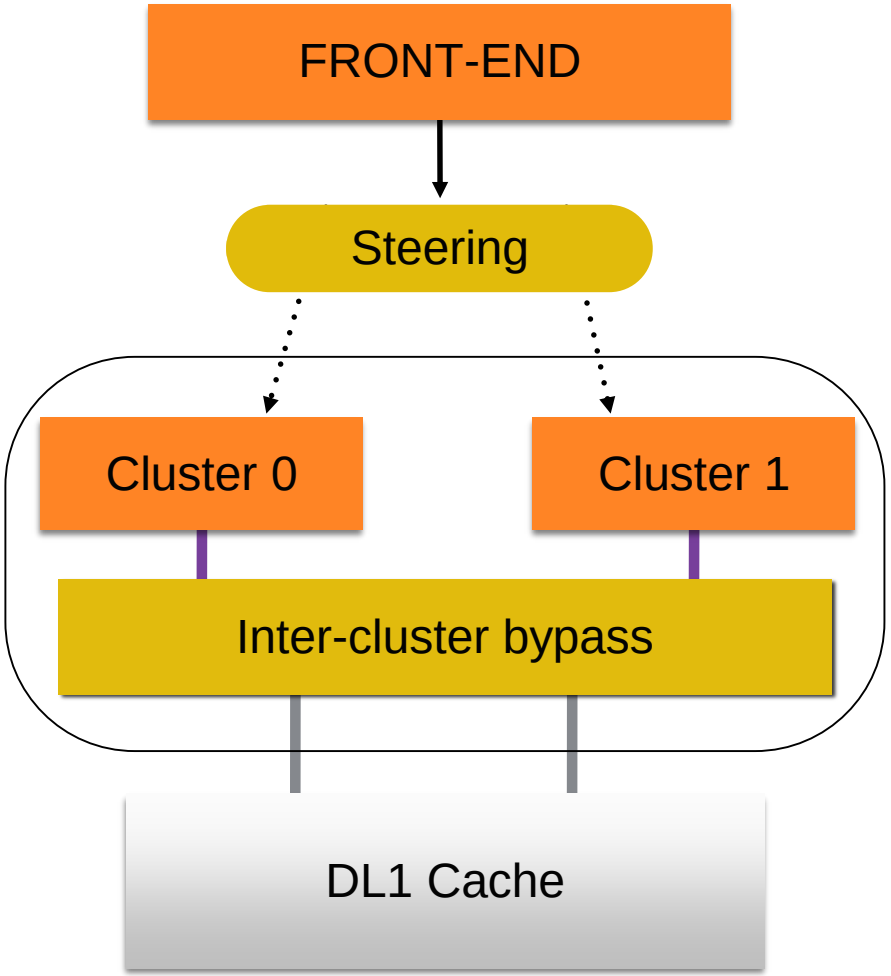
Future of Superscalar



Clustering + Register Write Specialization

Clustering


Divide
et
Impera

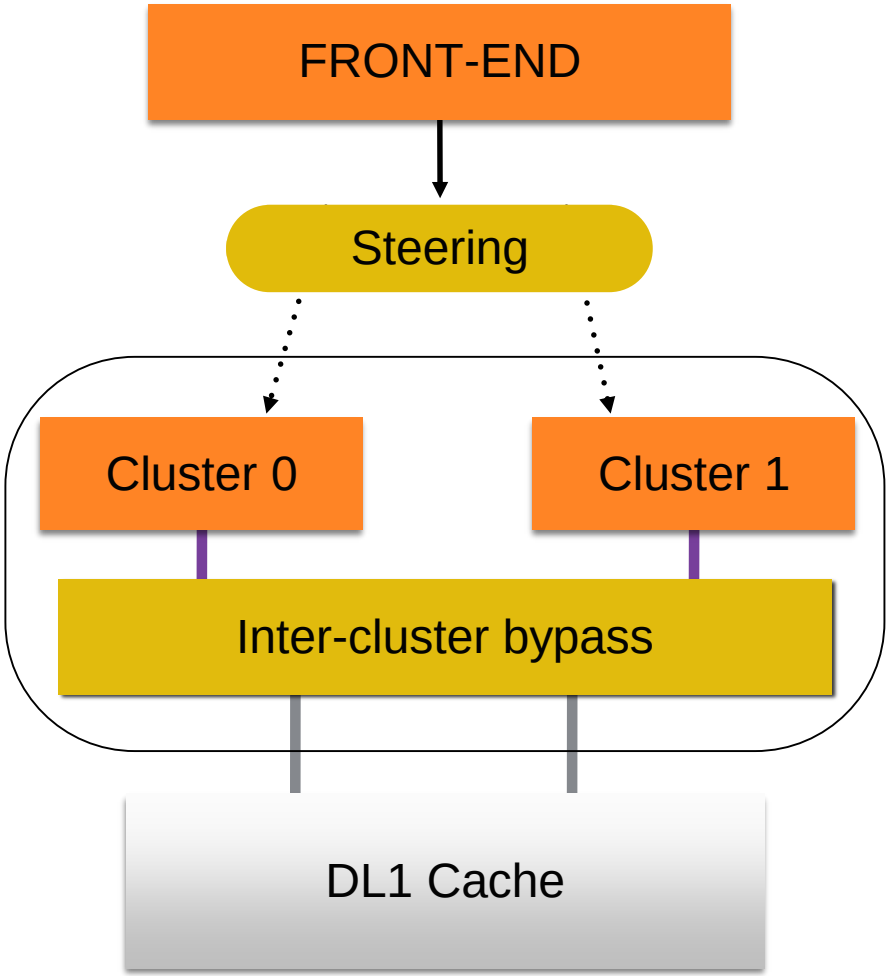

FRONT-END
Steering
Cluster 0
Cluster 1
Inter-cluster bypass
DL1 Cache

Clustering

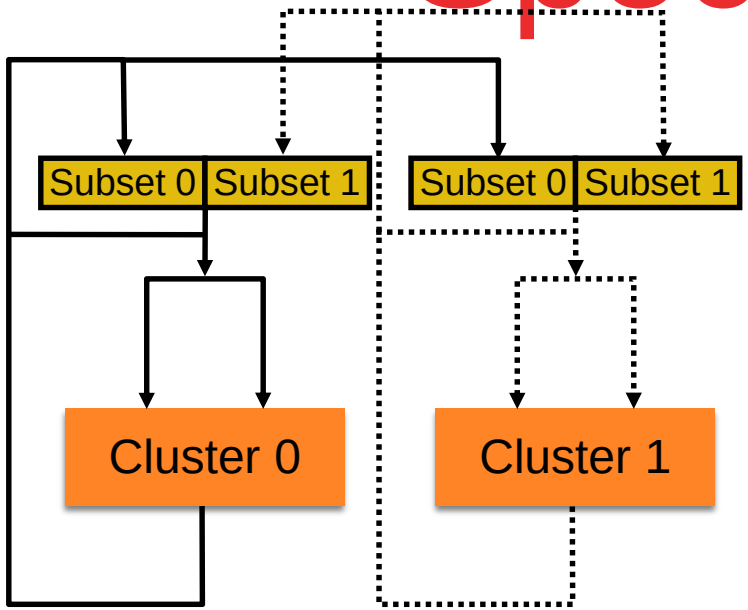
Steering Policy:

Mod-N

This scheme sends alternatively **N** instruction to each cluster



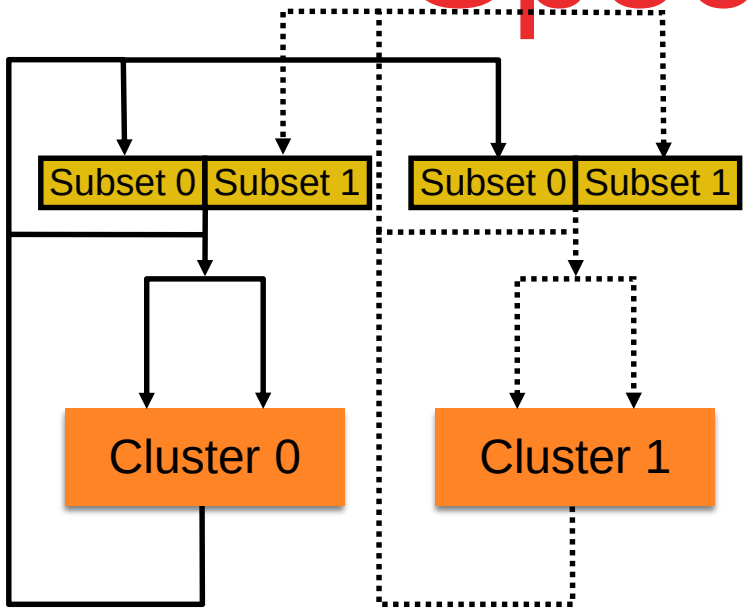
Register Write Specialization



Each cluster:

- *can write in its own register file partition*
- *has a mirror copy of the other register file partition*
- *we can keep the same numbers of write ports*

Register Write Specialization



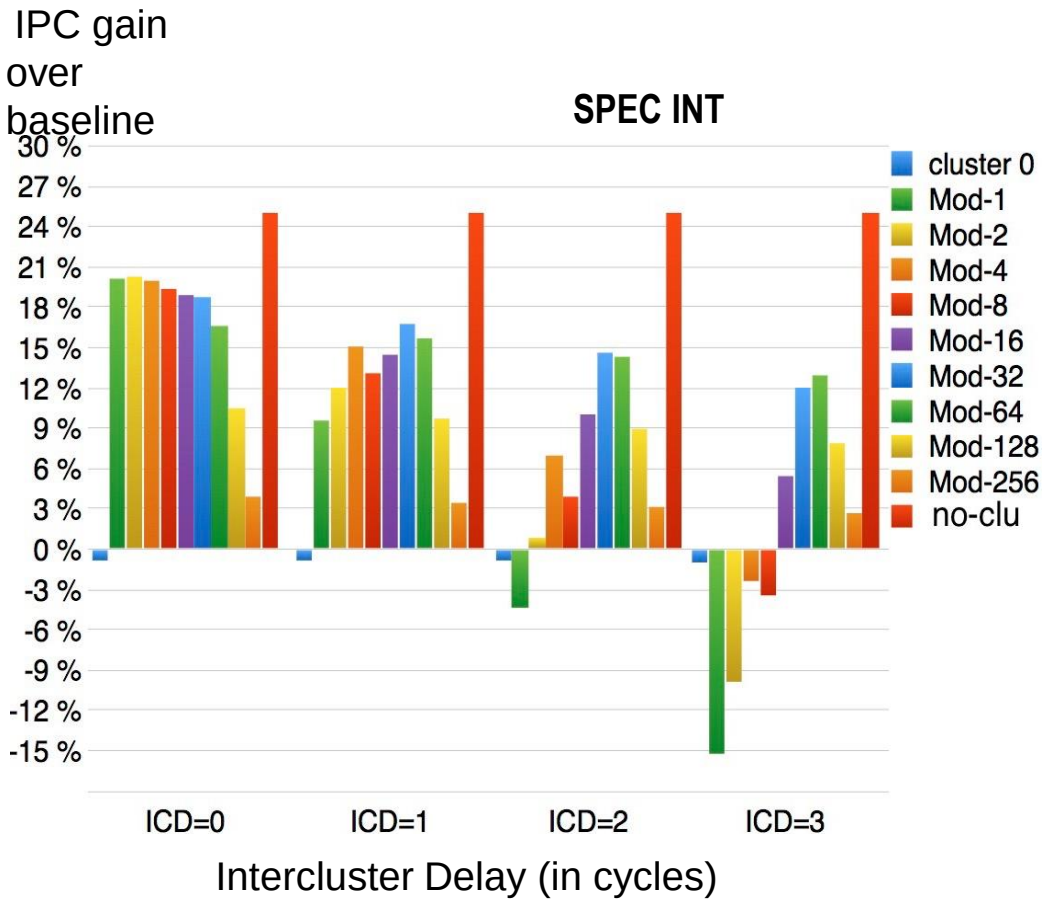
Each cluster:

- *can write in its own register file partition*
- *has a mirror copy of the other register file partition*
- *we can keep the same numbers of write ports*

steering should be finished before register renaming

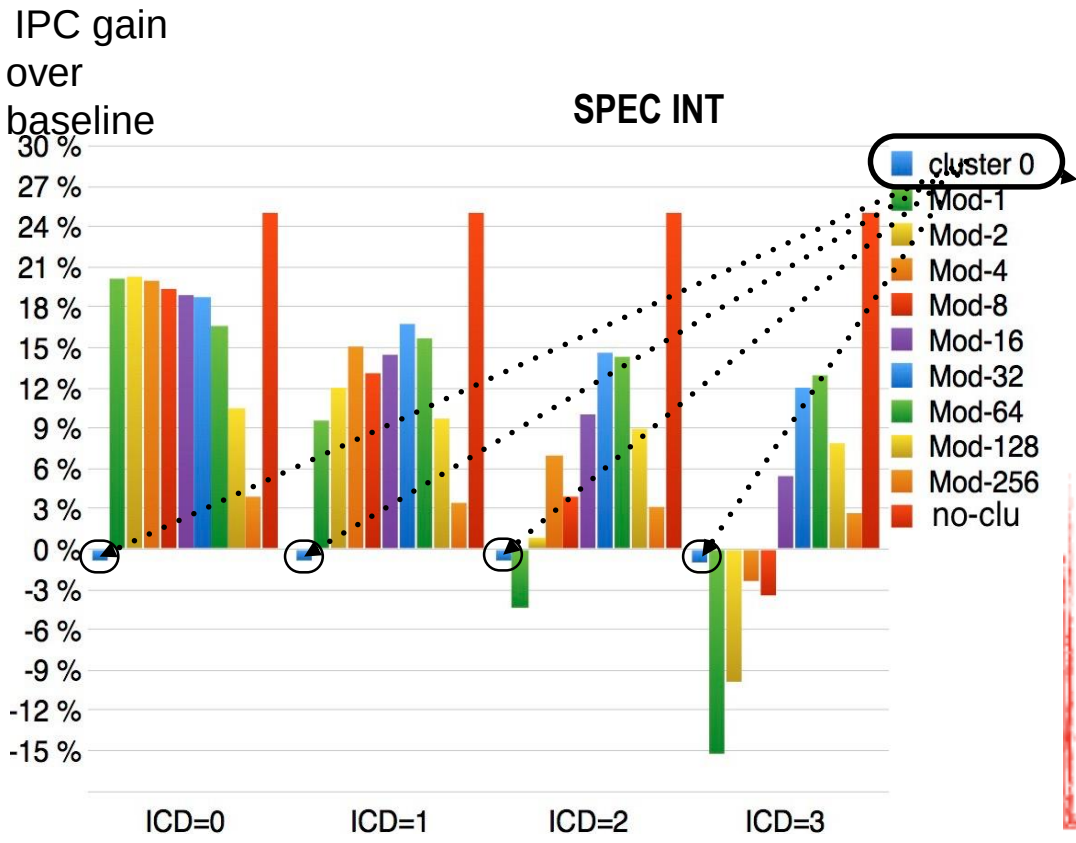
Register Write Specialization has no impact on final performances

Dual Cluster Performance



- *front-end doubled*
- *back-end doubled*
- *instruction window doubled*
- *DL1 latency from 3 to 4 cycles*

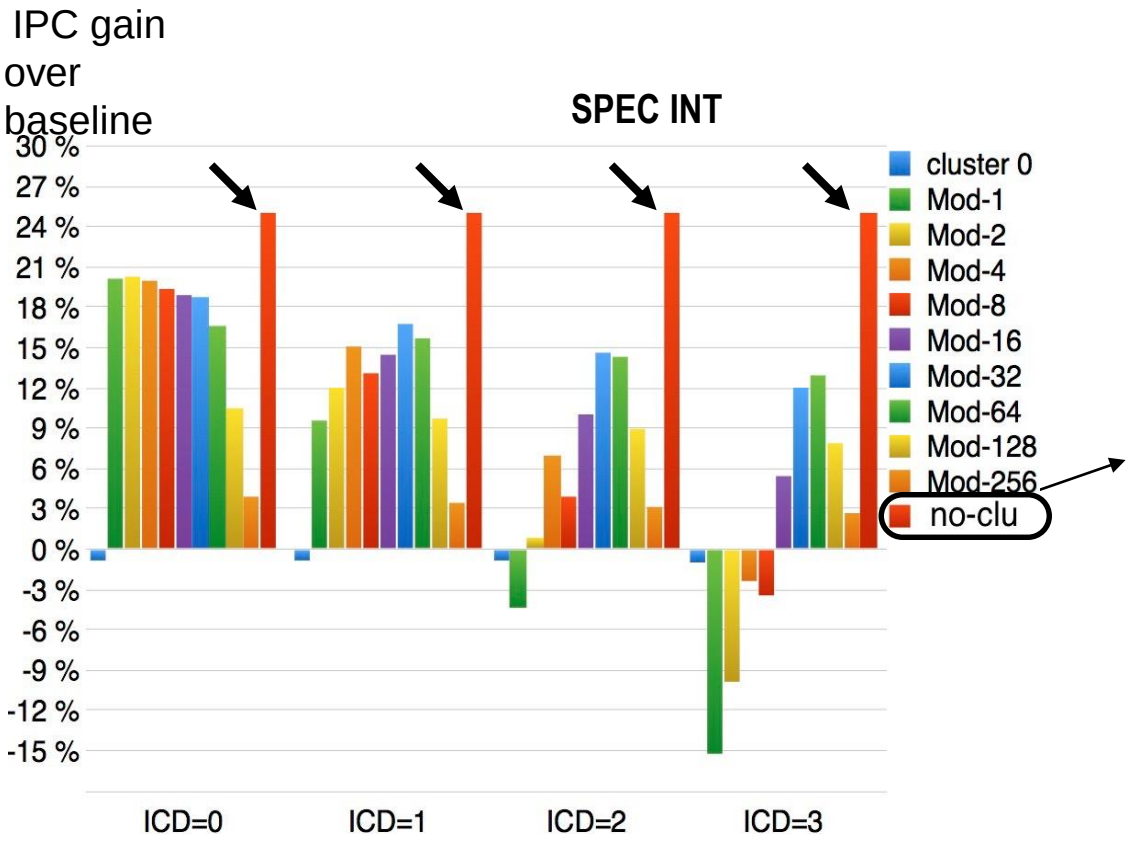
Dual Cluster Performance



steering all uops to cluster 0

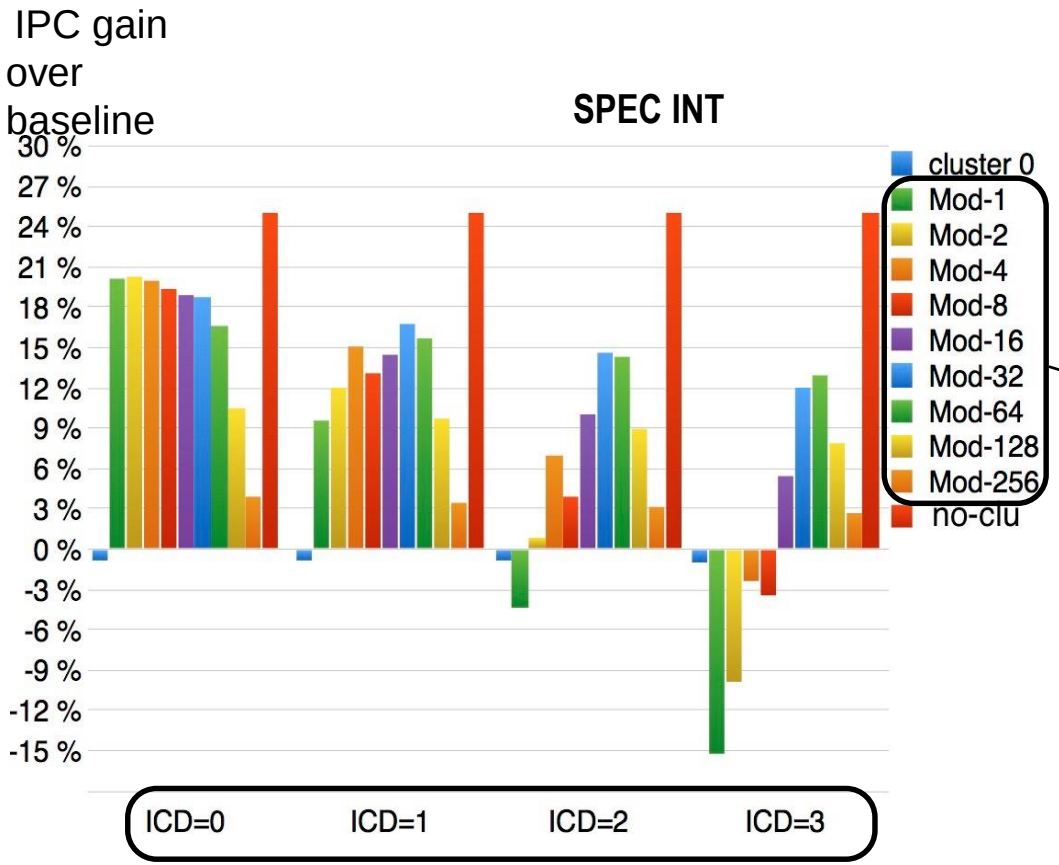
impact of DL1 compensated by ROB and MSHRs

Dual Cluster Performance



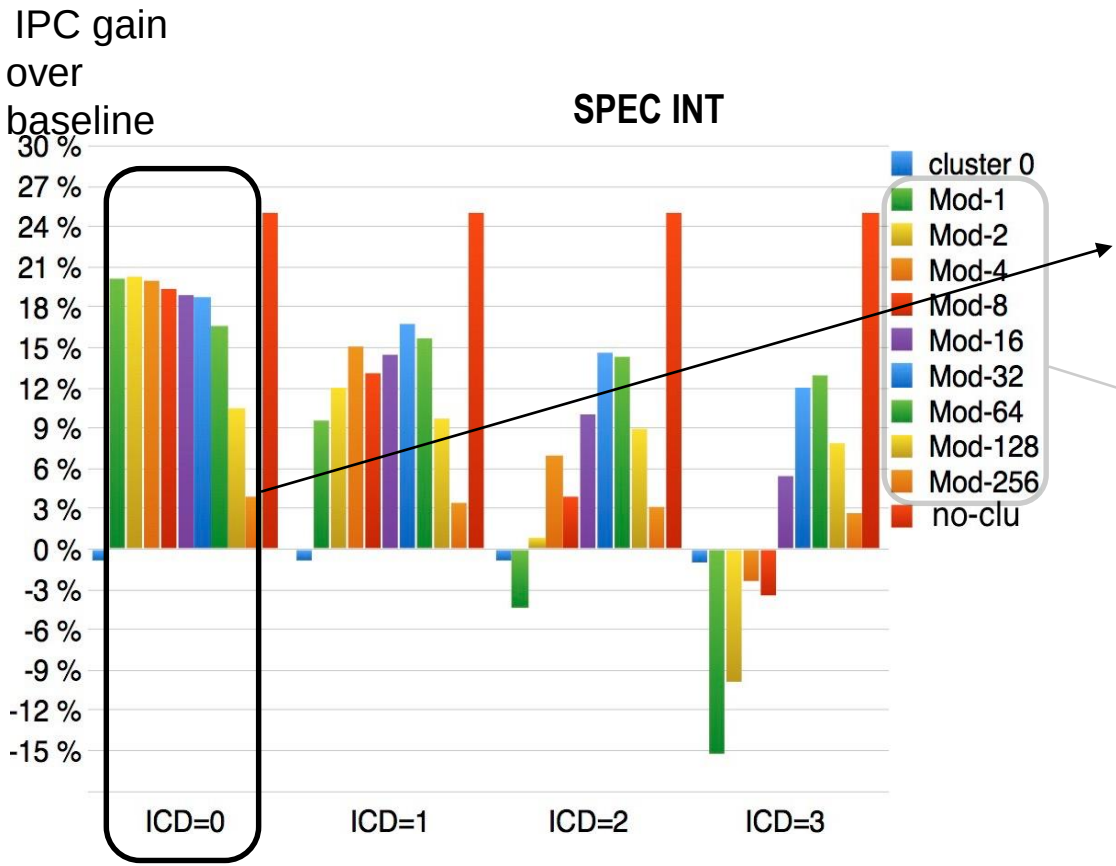
idealistic architecture

Dual Cluster Performance



incremental increase of Mod-N

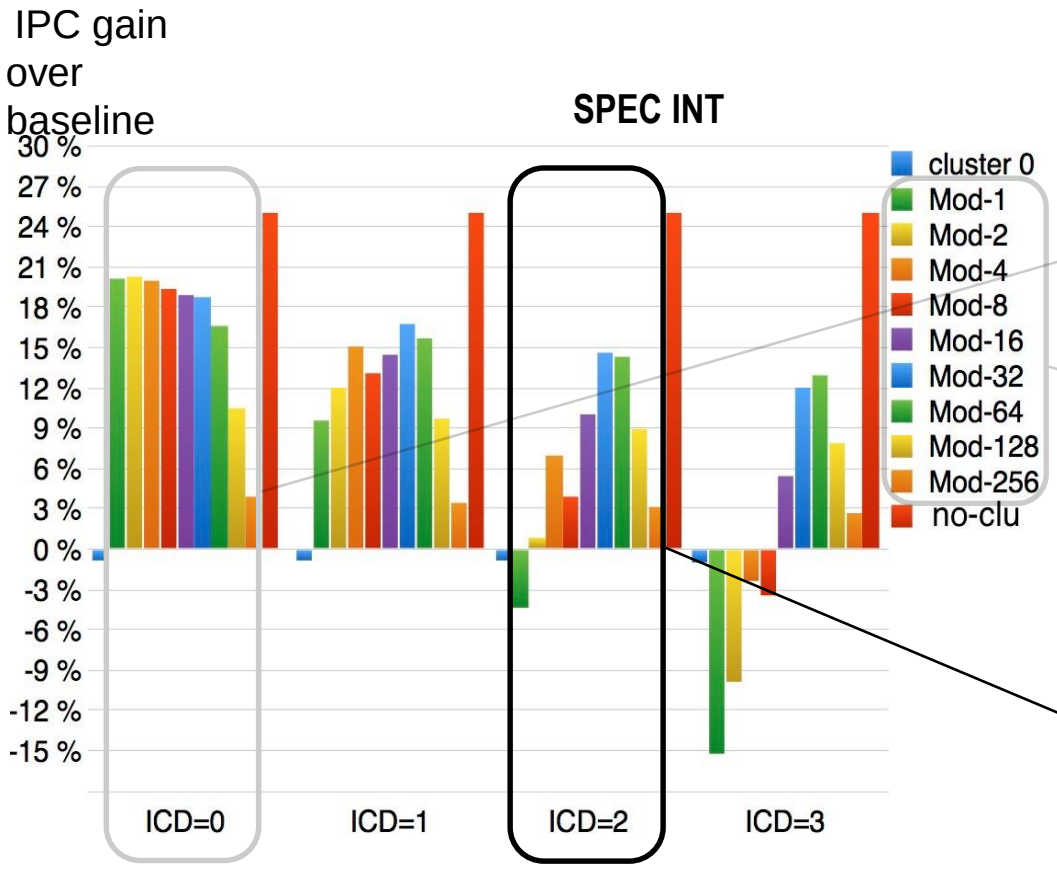
Dual Cluster Performance



N ≤ 32 achieves good balancing (ideal case)

incremental increase of Mod-N

Dual Cluster Performance



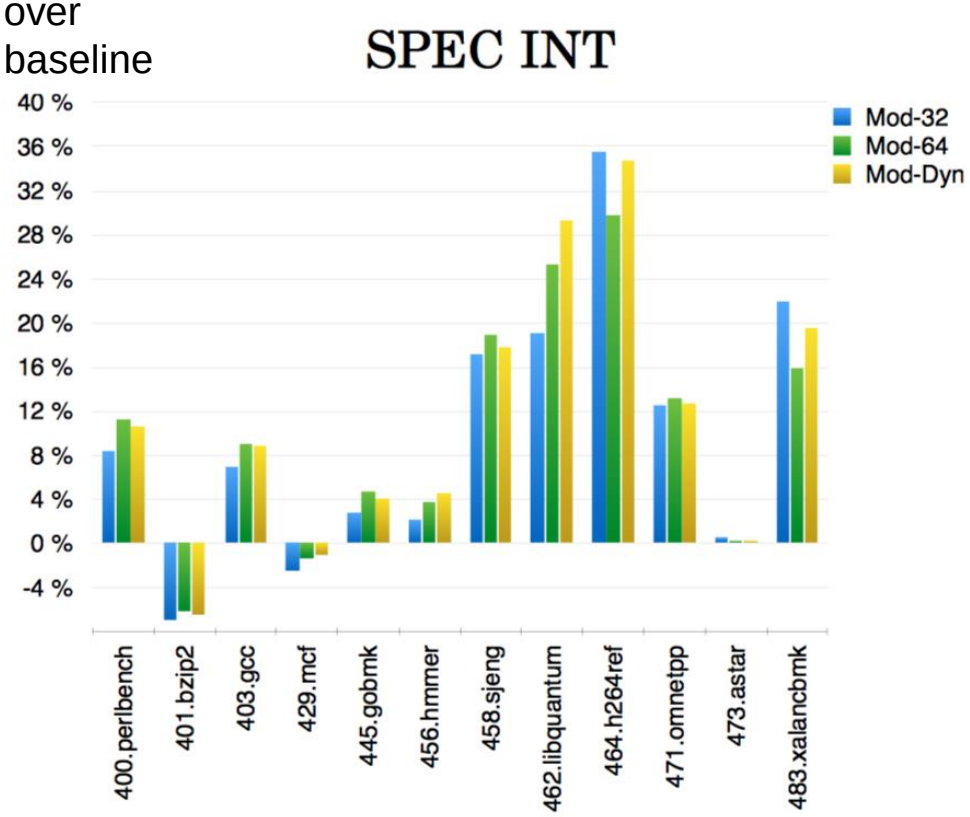
N ≤ 32 achieves good balancing (ideal case)

incremental increase of Mod-N

IPC very sensitive to Mod-N (realistic case)

Dual Cluster Perfomance

IPC gain over baseline



Intercluster Delay = 3 cycles

we could have an adaptive Mod-N

to find the best N dynamically

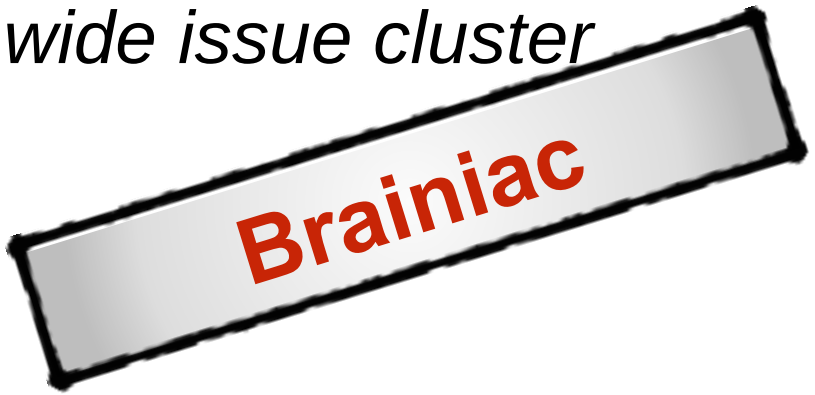
Clustering: a different approach

Past

- *increase clock freq*
- *narrow issue cluster*

Future

- *constant clock freq*
- *exploit more ILP*
- *energy efficiency*
- *wide issue cluster*



Clustering: Conclusions

Past

- *increase clock freq*
- *narrow issue cluster*

+

- *best steering MOD-3*

Future

- *constant clock freq*
- *exploit more ILP*
- *energy efficiency*
- *wide issue cluster*

+

- *best steering MOD-64*

Thank you

P. Michaud, A. Mondelli, A. Seznec, “Revisiting clustered microarchitecture for future superscalar cores: a case for wide-issue clusters“, ACM Transactions on Architecture and Code Optimization, Volume 12, Issue 3, August 2015.